1. What is API? Give 3 real-life examples of usage of API.

API:

API stands for ***Application Programming Interface***. In the context of APIs, the word Application refers to any software with a distinct functionality of interaction between two or more websites or apps.

Examples of usage of API:

a) When you search for weather on a website, an interface will ask you about a location. After putting a location it will fetch data from one or more weather report provider sites and display you a result. This is an example of using an API..
b) Need to book a flight. We google for a specific destination, google will display a form of information. After filling up the form, google will suggest different airlines with price and dates. This google form is an API.
c) Searching for a tool on your browser, browser will suggest you a specific website and will ask you search for it with a “Search” button within that business website (example: HomeDepot)

1. What is API testing and name some types of API testing?

API Testing:

In software application (app) development, API is the middle layer between the presentation (UI) and the database layer. APIs enable communication and data exchange from one software system to another.

API testing is a software testing practice that tests the APIs directly — from their functionality, reliability, performance, to security. Part of integration testing, API testing effectively validates the logic of the build architecture within a short amount of time.

API Testing Types

1. **Validation Testing**

Validation testing occurs among the final steps and plays an essential role in the development process. It verifies the aspects of product, behavior, and efficiency. In other words, validation testing can be seen as an assurance of the correct development.

2. **Functional testing**

Includes testing particular functions in the codebase. These features are the representation of specific scenarios to make sure the API functions are handled well within the planned parameters.

3. **UI testing**

UI testing is defined as a test of the user interface for the API and other integral parts. UI testing focuses more on the interface which ties into the API rather than the API testing itself. Although UI testing is not a specific test of API in terms of codebase, this technique still provides an overview of the health, usability, and efficiency of the app’s front and back ends.

4. **Security testing**

This practice ensures the API implementation is secure from external threats. Security testing also includes additional steps such as validation of encryption methodologies, and of the design of the API access control. It also includes user rights management and authorization validation.

5. **Load testing**

Load testing generally occurs after a specific unit or the whole codebase has been completed. This technique checks if the theoretical solutions work as planned. Load testing monitors the app's performance at both normal and peak conditions.

6. **Runtime and error detection**

This testing type is related to the actual running of the API — particularly with the universal results of utilizing the API codebase. This technique focuses on one of the below aspects: monitoring, execution errors, resource leaks, or error detection.

7. **Penetration testing**

Penetration testing is considered the second test in the auditing process. In this type, users with limited API knowledge will try to assess the threat vector from an outside perspective, which is about functions, resources, processes, or aim to the entire API and its components.

8. **Fuzz testing**

Fuzz testing is another step in the security audit process. In fuzz testing, a vast amount of random data (referred to as "noise" or "fuzz") will be input into the system to detect any forced crashes or negative behaviors. This technique tests the API's limits to prepare for the "worst-case scenarios."

3. What is the difference between REST and SOAP APIs?

REST was designed specifically for working with components such as media components, files, or even objects on a particular hardware device. Any web service that is defined on the principles of ***REST*** can be called a RestFul web service. A Restful service would use the normal HTTP verbs of GET, POST, PUT and DELETE for working with the required components. REST stands for Representational State Transfer.

SOAP is a protocol which was designed before ***REST*** and came into the picture. The main idea behind designing ***SOAP*** was to ensure that programs built on different platforms and programming languages could exchange data in an easy manner. ***SOAP*** stands for Simple Object Access Protocol.

KEY DIFFERENCE

- SOAP stands for Simple Object Access Protocol whereas REST stands for Representational State Transfer.
- SOAP is a protocol whereas REST is an architectural pattern.

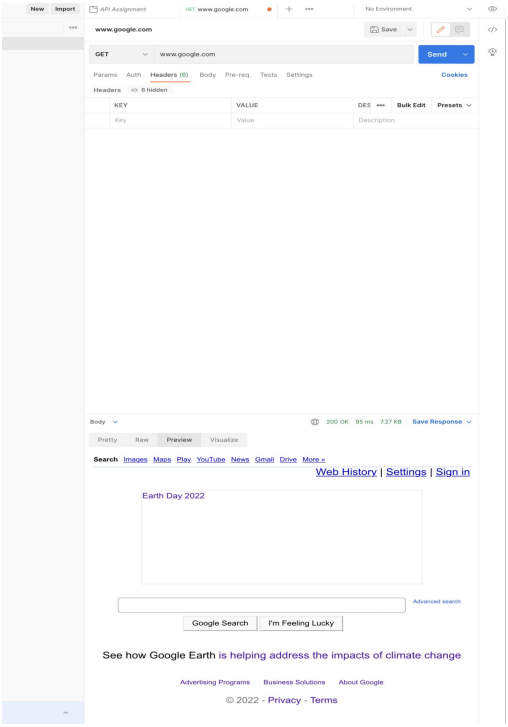
- SOAP uses service interfaces to expose its functionality to client applications while REST uses Uniform Service locators to access the components on the hardware device.
- SOAP needs more bandwidth for its usage whereas REST doesn't need much bandwidth.
- Comparing SOAP vs REST API, SOAP only works with XML formats whereas REST works with plain text, XML, HTML and JSON.
- SOAP cannot make use of REST whereas REST can make use of SOAP.
-

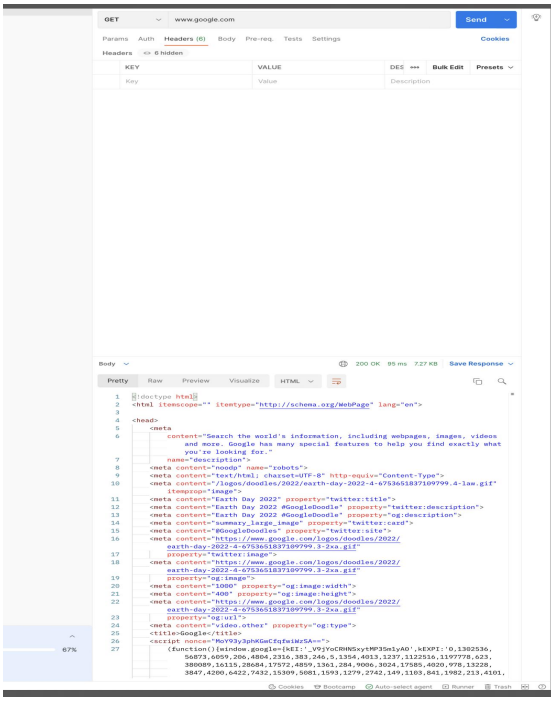
4. What are common HTTP methods? Explain with examples.

HTTP Methods:

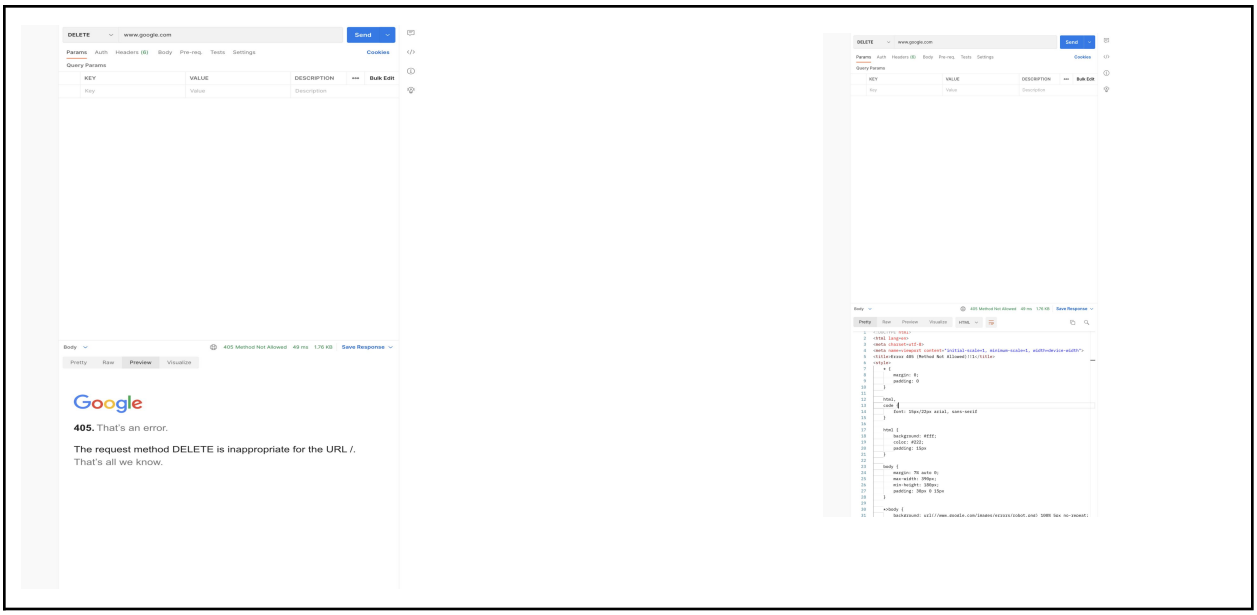
GET	The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.
HEAD	HEAD works same as GET, but transfers the status line and header section only.
POST	A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.
PUT	PUT Replaces all current representations of the target resource with the uploaded content.
DELETE	DELETE Removes all current representations of the target resource given by a URI.
CONNECT	CONNECT Establishes a tunnel to the server identified by a given URL.
OPTIONS	'OPTIONS' Describes the communication options for the target resource.
TRACE	TRACE Performs a message loop-back test along the path to the target resource.

Examples:
I have used www.postman.com to test the GET http method. My test subject was www.google.com. After sending a GET request, I received a response with html codes and preview of the page as follows.





But after requesting the DELETE http method, google responded with the message **“Method Not Allowed** “!

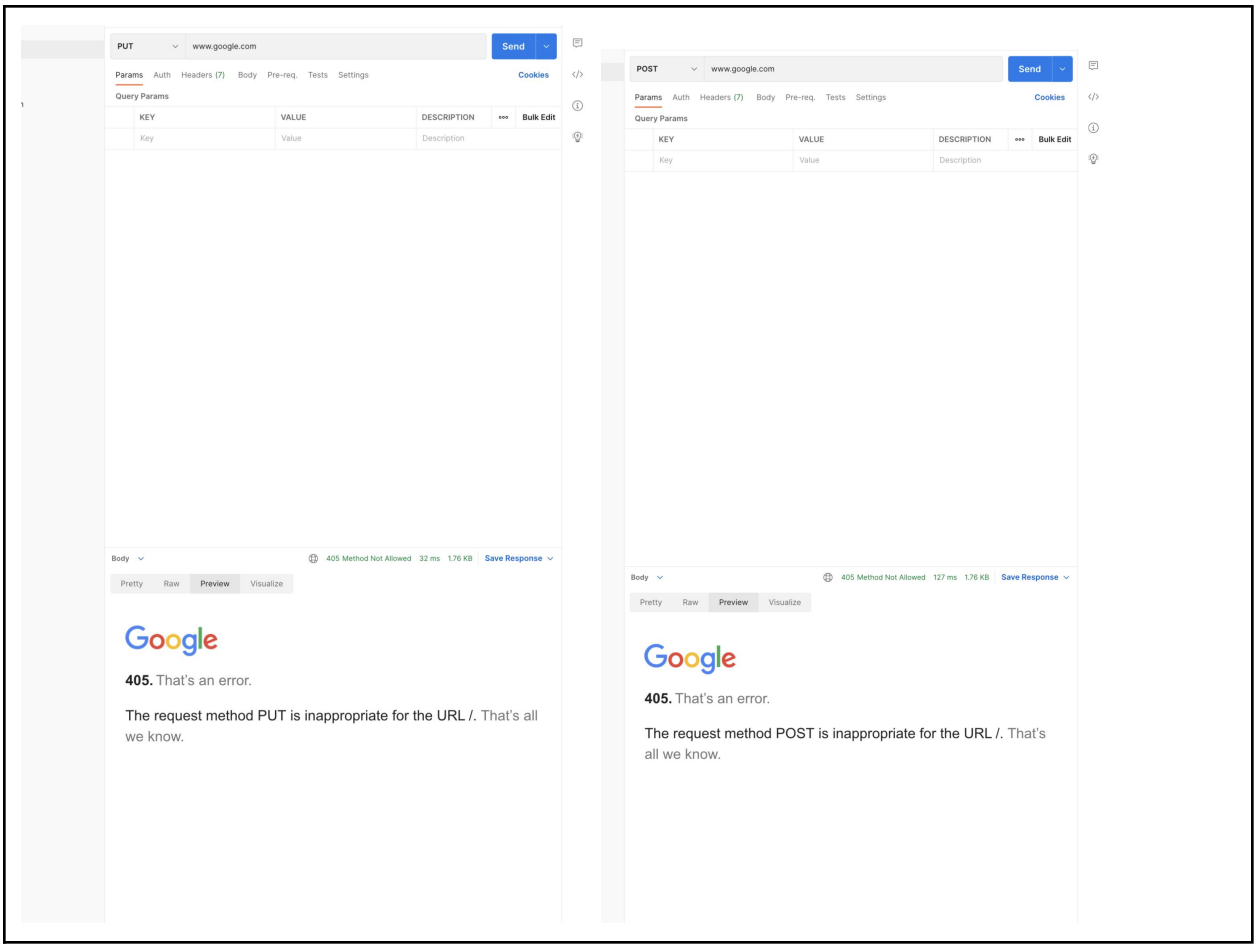


5. What is the difference between POST AND put? Explain with an example.

A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.

PUT Replaces all current representations of the target resource with the uploaded content.

As I run the same scenarios with the POST & PUT method, both times I have received messages saying the method is not appropriate.



6. Name 5 main categories of HTTP status codes? Explain 5 status codes from each category(You have to tell the status code and description of each status code with an example).

HTTP status codes are split into 5 different categories. Each category will give you hints as to what the response was, even if you don't know the specific response code.

5 main categories of HTTP status codes:

1. 1xx - Informational: The server has received the request and is continuing the process
2. 2xx - Successful: The request was successful and the browser has received the expected information
3. 3xx (Redirection): You have been redirected and the completion of the request requires further action

4. 4xx (Client Error): The website or the page could not be reached, either the page is unavailable or the request contains bad syntax
5. 5xx (Server Error): While the request appears to be valid, the server could not complete the request

Complete list of HTTP Status Codes

Status code Meaning

1xx Informational

100 Continue

101 Switching protocols

102 Processing

103 Early Hints

2xx Successful

200 OK

201 Created

202 Accepted

203 Non-Authoritative Information

204 No Content

205 Reset Content

206 Partial Content

207 Multi-Status

208 Already Reported

226 IM Used

3xx Redirection

300 Multiple Choices

301 Moved Permanently

302 Found (Previously "Moved Temporarily")

303 See Other

304 Not Modified

305 Use Proxy

306 Switch Proxy

307 Temporary Redirect

308 Permanent Redirect

4xx Client Error

400 Bad Request

401 Unauthorized

402 Payment Required

403 Forbidden

404 Not Found

405 Method Not Allowed

406 Not Acceptable
407 Proxy Authentication Required
408 Request Timeout
409 Conflict
410 Gone
411 Length Required
412 Precondition Failed
413 Payload Too Large
414 URI Too Long
415 Unsupported Media Type
416 Range Not Satisfiable
417 Expectation Failed
418 I'm a Teapot
421 Misdirected Request
422 Unprocessable Entity
423 Locked
424 Failed Dependency
425 Too Early
426 Upgrade Required
428 Precondition Required
429 Too Many Requests
431 Request Header Fields Too Large
451 Unavailable For Legal Reasons

5xx Server Error
500 Internal Server Error
501 Not Implemented
502 Bad Gateway
503 Service Unavailable
504 Gateway Timeout
505 HTTP Version Not Supported
506 Variant Also Negotiates
507 Insufficient Storage
508 Loop Detected
510 Not Extended
511 Network Authentication Required

---

5 important status codes:
HTTP Status Code 200 - OK. ...
HTTP Status Code 301 - Permanent Redirect. ...
HTTP Status Code 302 - Temporary Redirect. ...
HTTP Status Code 404 - Not Found. ...
HTTP Status Code 410 - Gone. ...
HTTP Status Code 500 - Internal Server Error. ...

HTTP Status Code 503 - Service Unavailable.

⁉⁉⁉⁉⁉⁉⁉

7. What are the main components of HTTP requests and HTTP responses?
HTTP works as a request-response protocol between a client and server. Example: A client (browser) sends an HTTP request to the server; then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

1. An HTTP request is made by a client, to a named host, which is located on a server. The aim of the request is to access a resource on the server. To make the request, the client uses components of a URL (Uniform Resource Locator), which includes the information needed to access the resource.
2. An HTTP response is made by a server to a client. The aim of the response is to provide the client with the resource it requested, or inform the client that the action it requested has been carried out; or else to inform the client that an error occurred in processing its request.

HTTP Response broadly has 3 main components:

- Status Line
- Headers
- Body (Optional)

8. What are the main components of the HTTP request header and response header?
HTTP headers let the client and the server pass additional information with an HTTP request or response.
An HTTP header consists of its case-insensitive name followed by a colon (:), then by its value. Whitespace before the value is ignored. A header composed of 3 headers: 1. Host, 2. Content-Type and 3. Cache-Control.

A request header is an HTTP header that can be used in an HTTP request to provide information about the request context, so that the server can tailor the response. For example, the Accept-* headers indicate the allowed and preferred formats of the response.

A response header is an HTTP header that can be used in an HTTP response and that doesn't relate to the content of the message. Response headers, like Age , Location or Server are used to give a more detailed context of the response.
HTTP Response broadly has 3 main components:1. Status Line. 2. Headers. 3. Body (Optional)

9. What is the difference between authentication and authorization?

Authentication vs. Authorization

Despite the similar-sounding terms, authentication and authorization are separate steps in the login process. Understanding the difference between the two is key to successfully implementing an Identity and Access Management (IAM) solution.

	Authentication	Authorization
What does it do?	Verifies credentials	Grants or denies permissions
How does it work?	Through passwords, biometrics, one-time pins, or apps	Through settings maintained by security teams
Is it visible to the user?	Yes	No
It is changeable by the user?	Partially	No
How does data move?	Through ID tokens	Through access tokens

10. Briefly explain common API Authentication Methods.

What is API Authentication?

Application Programming Interfaces (APIs)—the vital links that allow applications to exchange services and data—require authentication before the exchange can take place. If a client application tries to access another application, the target API wants to know: Is the client really the client it claims to be?

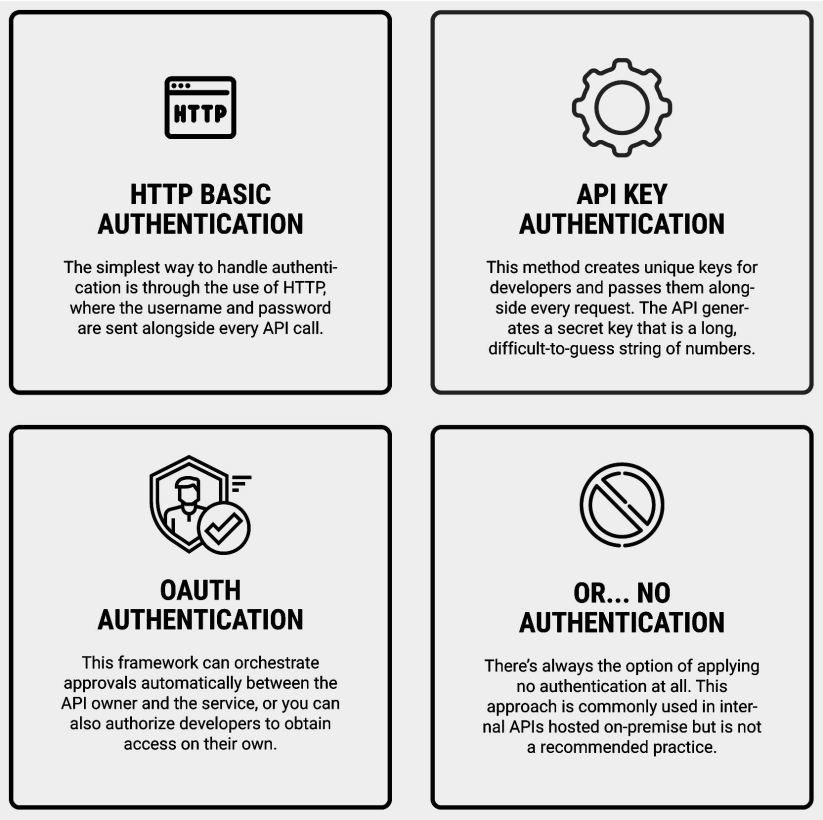
The API authentication process validates the identity of the client attempting to make a connection by using an authentication protocol. The protocol sends the credentials from the remote client requesting the connection to the remote access server in either plain text or encrypted form. The server then knows whether it can grant access to that remote client or not.

That's essentially what API authentication is all about. The system needs to make sure each end-user is properly validated. The system also wants to make sure the client system does not

represent someone who has accidentally tried to access a service they are not entitled to, or worse—the system of a cybercriminal trying to hack into the system.

Common API Authentication Methods

There are a variety of ways to authenticate API requests. Here are the three most common methods:



API Authentication Methods:

HTTP Basic Authentication
The simplest way to handle authentication is through the use of HTTP, where the username and password are sent alongside every API call. You can use an HTTP header and encode the username and password. Note that does not mean . If you end up using HTTP Basic Authentication, use it through HTTPS so the connection between the parties is encrypted.

API Key Authentication
This method creates unique keys for developers and passes them alongside every request. The API generates a secret key that is a long, difficult-to-guess string of numbers and letters—at least 30 characters long, although there's no set standard length. It is typically passed alongside the API authorization header.

OAuth Authentication

For HTTP services, you can give third-party developers access by using the OAuth 2.0 authorization framework. This framework can orchestrate approvals automatically between the API owner and the service, or you can also authorize developers to obtain access on their own.

No Authentication

There's always the option of applying no authentication at all. Developers can just make a request to a specific URL and get a response without needing any credentials or an API key. This approach is commonly used in internal APIs hosted on-premises but is not a recommended practice.